Exerciseur

Clemente Paredes

Compilation et Éxecution:

Pour compiler l'exerciseur, utiliser la commande:
`gcc exer.c -o ex -lm`. Alternativement, un fichier déjà compilé `ex` est inclut.

Pour executer l'exerciseur compilé, utiliser la commande:
`./ex`

Le programme va ensuite demander si l'utilisateur veut génerer des expressions arithmétiques ou des polynômes de second degré.

Choix d'implementation:

Géneration d'Expressions Arithmétiques:

La géneration d'Expressions Arithmétiques commence dans la fonction `char* expression` qui renvoie une chaîne de caractères avec une expression arithmétique. Elle prends pour paramètres les `int n, d, lvl`.

- `n` désigne le nombre de sous-calculs par expression.
- `d` désigne le nombre maximale de chiffres par nombre.
- `lvl` désigne la 'difficulté' de l'opération. 0 indique que le résultat de l'expression sera positif, ainsi que celui de toutes les sous-operations. 1 indique l'opposé.

Ces paramètres sont demandés en saisie par la fonction intéractive `void arith_interac()`. La fonction génère un chiffre aléatoire qui, selon sa valeur, désigne l'opération `(+, -, *, /)` à ajouter a l'expression finale. Ceci est aussi fait pour lier les sous-opérations entre elles avec des opérateurs. Des parenthèses ont aussi été implementes, qui peuvent être inclus dans l'expression aléatoirement. Ils sont correctement fermés à la fin de l'execution du programme.

Si `lvl == 1`, des opérations sont crées jusqu'en obtenir une où toutes les sous-operations donnent un résultat positif. Les divisions (et donc aussi les expressions finales) ont toujours des résultats entiers.

La fonction `void savefile()`, prenant par argument `n, d, lvl`, ainsi que `k` (aussi demandé en saisie), qui désigne le nombre d'expressions à garder, sauvegarde les expressions dans le fichier `expressions.txt`.

Pour en évaluer ces expressions et sauvegarder les résultats, ainsi que pour tester les sous-opérations, on utilise la calculatrice présente dans le dossier `infixpostfix`. Son fonctionnement à déjà été expliqué. Ici on l'utilise avec la fonction `system()`.

`system("cd infixpostfix && echo "[`**EXPRESSION**`]> ../temp.txt")`

Le résultat est sauvegardé dans un fichier temporel `temp.txt` si l'évaluation est temporelle, ou en `expanswers.txt` si c'est les résultats des expressions finales. Les espaces `(" ")` des expressions sont toujours éliminés avant les évaluer, pour éviter des erreurs avec le programme `infixpostfix`.

Géneration d'Équations Quadratiques.

La géneration d'Équations Quadratiques suit une logique similaire à celle de la géneration d'expressions arithmétiques. Trois entiers, `a, b, et c`, correspondant aux coefficients du polynôme ax^2+bx+c sont établis aléatoirement, aussi avec une quantité de chiffres posé en paramètre. Les résultats en `expanswers.txt` vont être les racines du polynôme. On veut avoir des racines réelles et entières. On utilise alors la proprieté suivante:

Un polynôme ax^2+bx+c admet des racines entières si et seulement si
1. b^2-4ac est un carré parfait ou entier
2. a divise b et c

On génère alors la quantité demandé de polynômes, en les gardant seulement si ces conditions sont bien suivies. Une fonction `isperfsquare(int d)` est aussi utilisé pour verifier le discriminant. Les deux solutions sont ensuite calculés avec la formule:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Si les deux solutions sont égales ($b^2 - 4ac = 0$), la racine doublé est consideré elle seule.
Les racines de chaque polynôme gardé en `expressions.txt` sont sauvegardés en `expanswers.txt`

Afficher:

L'exerciseur peut aussi afficher les fichiers de texte crées, avec la commande `system("cat expressions.txt`

Ouverture:
Améliorations possibles:

Plusieurs des fonctions en charge de vérifier les conditions basiques des expressions prennent largement de temps à partir d'un nombre élevé de sous-calculs ou de chiffres par nombre. Des optimizations à ces fonctions peuvent améliorer cela. Une structure de données plus avancée comme un arbre binaire est un example d'emploie plus optimale.

Possibles ajouts à futur:

L'exerciseur pourrait aussi permettre à l'utilisateur de résoudre interactivement les expressions générés, en comparant les résultats saisies avec ceux présents en `expanswers.txt`